CLAIMS

What is claimed is:

1. A method of operating a computer system, said computer system including at least one processor, comprising.

establishing a plurality of memory units each having a corresponding memory location;

executing a plurality of tasks running on said at least one processor, said plurality of tasks being operable to share data;

defining a plurality of lists for each memory location;

determining the validity of said data in said memory unit;

locking at least one of said plurality of lists if said data is invalid;

inserting an entry corresponding to one of said plurality of tasks onto said locked list;

unlocking said locked list; and

determining if data is inputted in said memory location between said determining step and said unlocking step.

2. A method for operating a computer system, said computer system comprising at least one process, comprising:

establishing a plurality of memory units each having a corresponding memory location;

running a plurality of tasks on said processor, said plurality of tasks being operable to share data;

defining a plurality of lists for each memory location;

inserting an entry corresponding to one of said plurality of tasks onto one of said plurality of lists if said one list is unlocked; and

determining if another of said lists is unlocked if said one list is locked.

3. A method for synchronizing processes in a computer system, said computer system including at least one processor, comprising:

establishing a plurality of memory units each having a corresponding memory location

executing a plurality of tasks running on said processor, said plurality of tasks being operable to share data located in said memory units;

defining a plurality of lists for each memory location;

locking at least one of said plurality of lists if said data is not valid;

inserting an entry corresponding to one of said plurality of tasks onto said locked list;

unlocking said locked list;

suspending said entered task until valid data is found in said memory unit;

reading said valid data;

determining if other data is inputted in said memory unit before said locking step and after said unlocking step; and

reading said other data if it appears in said memory unit.

4. The method of claim 3, wherein the locking step further comprises activating selected other ones of said plurality of tasks that are entered on said locked list.

5. The method of claim 3, wherein said plurality of lists forms a linked list.

6. The method of claim 3, wherein said plurality of lists is between four and eight.

7. The method of claim 3, further comprising transferring the operation of said locked list when said locked list is locked by another one of said plurality of tasks.

8. A computer system having enhanced concurrency, comprising:

a plurality of processors;

a plurality of tasks running on said plurality of processors;

a plurality of memory units each having a corresponding memory location;

a plurality of lists corresponding to each of said memory location;

wherein one of said plurality of tasks is responsible for activating selected ones of said plurality of tasks contained on the same list as said one task.

9. The system of claim 8, wherein said plurality of lists form a linked list.

10. The system of claim 8, wherein said plurality of lists is between four and eight.

11. The system of claim 8, wherein said computer system is a multitasking or multiprocessing computer system.

12. A method of operating a computer system having at least one processor, comprising:

determining the validity of data in a memory unit;

locking a list corresponding to said memory unit if said data is invalid;

inserting an entry corresponding to one of said plurality of tasks onto said locked list;

unlocking said locked list; and

determining if data is inputted in said memory location between said determining step and said unlocking step.

13. A method for performing an operation within limits upon a shared value stored in an actual value location comprising:

providing a plurality of memory locations, two of said memory locations being first and second reservation memory locations and two of said memory locations being limit memory locations containing limit values;

getting an operand;

performing the operation upon an affected reservation register using the operand;

comparing a resulting value of the first operation upon the affected reservation register to the limit values in the limit memory locations;

if the resulting value is not within the values in the limit memory locations, then performing an inverse operation to restore the affected reservation register and reporting a failure; and

if the resulting value is within the values in the limit memory locations, then performing the operation to update the actual value location, performing the operation to update an unaffected reservation register, and report a success.

14. A system for performing an operation within limits upon a shared value stored in an actual value location comprising:

a plurality of memory locations, two of said memory locations being first and second reservation memory locations, and two of said memory locations being limit memory locations containing limit values.